A GEOMETRIC ICA PROCEDURE BASED ON A LATTICE OF THE OBSERVATION SPACE

M. R. Álvarez [1], F.Rojas[1], C.G. Puntonet[1], J.Ortega[1], F.Theis[2], E.W.Lang[2]
[1] Dept. Architecture and Computer Technology. Univ. Granada (Spain)
[2] Institute of Biophysics. Univ. Regensburg (Germany)

ABSTRACT

This work shows a new method for blind separation of sources, based on geometrical considerations concerning the observation space. This new method is applied to a mixture of several sources and it obtains the estimated coefficients of the unknown mixture matrix A and separates the unknown sources. The principles of the new method and a description of the algorithm followed by some speed enhancements are shown. Finally, we illustrate with simulations of several source distributions how the algorithm performs.

1. INTRODUCTION

The separation of independent source signals from mixed observed data is a fundamental and challenging signal processing problem. In many practical situations, one or more desired signals need to be recovered blindly knowing only the observed sensor signals. When p different source signals propagating through a real medium have to be captured by sensors, these sensors are sensitive to all sources $s_i(t)$ and thus the signal $x_k(t)$, observed at the output of sensor *k*, is a mixture of source signals. With a linear and stationary mixing medium the sensor signals can be described by:

$$\vec{x}(t) = A\,\vec{s}(t) \tag{1}$$

where $\vec{x}(t) = (x_1(t),\ ...,\ x_n(t))^T$ is an experimentally observable $(n\times 1)$-sensor signal vector s(t), with $\vec{s}(t) = (s_1(t),\ ...,\ s_p(t))^T$ is a $(p\times 1)$- unknown source signal vector having stochastic independent and zero-mean non-Gaussian elements $s_i(t)$, and *A* is a $(n\times p)$ unknown full-rank and non-singular mixing matrix. The solution of the blind signal separation (BSS) problem consists of retrieving the unknown sources $s_i(t)$ from just the observations. To achieve this it is necessary to apply the hypotheses that the sources $s_i(t)$ and the mixture matrix $A = (\vec{a}_1,...,\vec{a}_n)^T$ are unknown, that the number *n* of sensors is at least equal to the number *p* of sources, i.e. $n \geq p$, and that the components of the source vector are statistically independent yielding:

$$p(\vec{s}) = \prod_{i=1}^{n} p(s_i) \tag{2}$$

In order to solve the BSS - problem a separating matrix W is computed whose output is an estimate of the vector $\vec{s}(t)$ of the source signals (Figure 1) such that:

$$\vec{y}(t) = W\ \vec{x}(t) \tag{3}$$

Any BSS algorithm can only obtain W subject to:

$$W \cdot A = DP \tag{4}$$

with a diagonal scaling matrix *D* modified by a permutation matrix *P*. Recently, blind source separation (BSS) and Independent Component Analysis (ICA) have received much attention because of its potential applications in signal processing. A great diversity of estimation methods have been proposed based on some kind of statistical analysis, neural networks [7], the entropy concept [3], the geometric structure of the signal spaces [1,6,9], the fixed-point algorithm FastICA [5], the maximum likelihood stochastic gradient algorithm [2], the Jade algorithm [4], among others. Several geometric procedures have been used to separate either multivalued or analog signals, by analyzing the observed sensor signals in the resulting p-dim space of observations. In the following we will present a new geometric ICA algorithm which is based on rough density estimation.

2. PRINCIPLES OF THE NEW METHOD.

For *p = 2* and with bounded values (uniform distribution), the observed signals $(x_1(t),\ x_2(t))$ form a parallelogram in the $(\vec{x}_1, \vec{x}_2)$ space, as shown in Figure 1. We have demonstrated [8] that, through a matrix transformation, the coefficients of the matrix coincide with the slopes of the parallelogram. It can be seen that for random uniform sources, the parallelogram representing the space of observations $(\vec{x}_1, \vec{x}_2)$ is geometrically bounded within the

segments between the points P_1 to P_4. The slopes of these segments give the coefficients of the estimated mixture matrix W^{-1}. In order to obtain these segments, it is necessary to estimate the coordinates of those points P_i., $_{i=1...4}$. Assuming non-uniformly distributed signal as the sources, for example speech signals with an underlying super-Gaussian distribution; the form of the sensor signal distribution in the space of observations is highly nonuniform too, as can be seen in Figure 5. In this case it is not sufficient to estimate the borders of the bounded space of observations. Rather, it is necessary to detect the directions of high density in the space of observations.

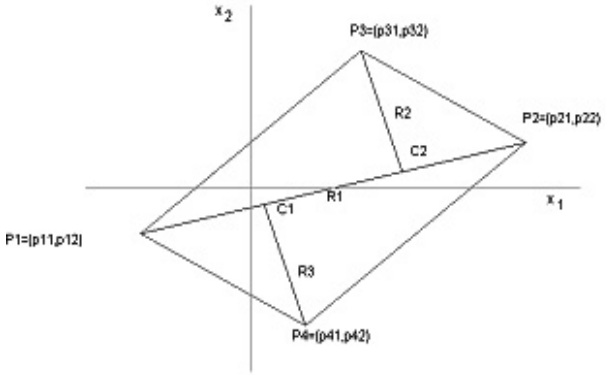

Fig. 1. Space of observations: Representative points and straight lines.

2.1. Description of the algorithm.

First of all, the algorithm computes the kurtosis of each component of the sensor signals and also the correlation coefficients between all observations. This is to detect whether the underlying source signal distributions correspond to sub- or super-Gaussian distributions. According to the Central Limit Theorem, mixtures will tend to be closer to Gaussian than the original ones. Consequently, kurtoses of the mixtures will be closer to zero (Gaussian distribution) than the sources:

$$\left|Kurt(x_i)\right| \le \max\left\{ \left|Kurt(s_j)\right| \right\} \quad i,j \in [1...n]$$

In any case, for mixtures of two signals, they will tend to preserve the sub- or super-Gaussian nature of the original signals, assuming that both sources have the same sign in the kurtosis. If the kurtoses of all sensor components are positive, the algorithm searches for high density regions of the sensor signal distribution. With sub-Gaussian signals, the algorithm estimates the bounding box of the parallelogram representing the space of observations. The algorithm subdivides the space of observations $(\vec{x}_1, \vec{x}_2)$ into a regular lattice of cells with N-rows and M- columns (lattice of N by M) as shown in Figure 2. Then, the algorithm computes the number of cells in the lattice in which the number of points inside it is greater than a given threshold TH.

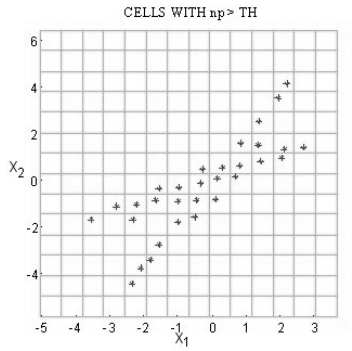

Fig. 2 - Lattice of the space of observations in cells

The distribution of sensor signals within each of these cells then is replaced by a prototype sensor signal vector. The prototype vector mostly does not point towards the centre of the cell because its position is weighted by the density of points (x_{1i}, x_{2i}) in this cell. This step greatly reduces the complexity of the algorithm, because the greatest number of points that the procedure needs to compute is $N \times M$ (if the space of observations contains 10^5 points (x_1, x_2) and *N=M=20* then the procedure only requires $N \times M = 400$ points. To further reduce the number of points, the next step of the algorithm finds those points which either form the border of the hyperparallelepiped or mark the high density regions of the sensor signal distribution in the space, by looking for cells that have an empty neighborhood (such cells have fewer points than the threshold TH). Then these cells without a complete neighborhood form the border of the distribution encompassing NR data points in the space of observations. The algorithm then computes the coordinates of $P_1 = (p_{11}, p_{12})$ and $P_2 = (p_{21}, p_{22})$. The space of observations has been reduced to NR data points which, in two dimensions, represent pairs of coordinates (x_{1i}, x_{2i}). In this reduced set of NR data points, there exist data points P_1 and P_2 with largest Euclidean distance between them in the space of observations:

$$d(P_1, P_2) = \max_{i,j \in (1,2,....NR)} d(P_i, P_j) \tag{5}$$

Once points P_1 and P_2 have been identified, the algorithm calculates the equation of the straight line R_1 which passes through these points P_1 and P_2 :

$$Ax_1 + Bx_2 + C = 0 \tag{6}$$

where:

$$A = (p_{22} - p_{12}),\ B = (p_{11} - p_{21}),$$
$$C = (p_{21} - p_{12}) - (p_{22} - p_{11}) \quad (7)$$

Next, the algorithm estimates the coordinates of the points $P_3 = (p_{31}, p_{32})$ and $P_4 = (p_{41}, p_{42})$ as follows: the straight line R_1 divides the space of observations $(\vec{x}_1, \vec{x}_2)$ into two subspaces, being R_1 the border between them. Data points which lie within one of these subspaces yield a nonzero result in Eq. (6). For example, data points lying above the straight line R_1 yield a negative result in Eq. (6). There is then one data point $P_3 = (p_{31}, p_{32})$ which provides the most negative value of all possible outcomes of Eq. (6), hence which also represents the point with the greatest Euclidean distance from the straight line R_1 in the subspace above R_1. In the same way, points in the other subspace, below the straight line R_1, yield a positive result in Eq. (6). Again, there is one point $P_4 = (p_{41}, p_{42})$ that provides the most positive value of all possible results from Eq. (6), and which is also the point with greatest Euclidean distance from the straight line R_1 in the subspace below R_1. In both cases, the algorithm calculates the Euclidean distance $r(P_i)$ from a generic point $P_i = (p_{i1}, p_{i2})$ to the straight line R_1.

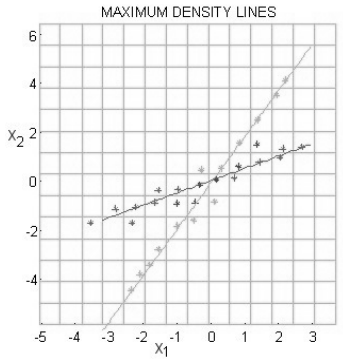

Fig. 3. Straight lines which define the separation matrix.

Once the characteristic points of the parallelogram have been obtained, the algorithm computes either the slopes of the segments ($\overline{P_1P_3}$ and $\overline{P_1P_4}$ or, equivalently $\overline{P_2P_4}$ and $\overline{P_3P_2}$) in case of sub-Gaussian densities or the slopes of the diagonals ($\overline{P_1P_2}$ and $\overline{P_3P_4}$) in case of super-Gaussian densities in order to obtain the coefficients of the matrix W as in Eq. (8) (see Figure 3):

$$\left(\frac{a_{12}}{a_{22}}\right)^{-1} = \frac{p_{32} - p_{12}}{p_{31} - p_{11}};\left(\frac{a_{21}}{a_{11}}\right) = \frac{p_{42} - p_{12}}{p_{41} - p_{11}} \quad (8)$$

Using the coefficients of matrix W, the algorithm computes the inverse matrix W^{-1} and reconstructs the unknown source signals $\vec{s}(t)$ (see Eq. (3)).

2.2. Further enhancements.

The computational order of the algorithm is polynomial:

$$Comput - Order = (DataPoints^2 \cdot XColumns \cdot YRows) \quad (9)$$

As a further improvement, we propose the reduction of the number of points at the beginning of the algorithm with a random elimination through all the space of the joint distribution of the mixtures as long as enough data points are kept to correctly estimate the sources. A more elaborated proposal is eliminating those points of the joint distribution of the mixtures which lay within a calculated radius near the center of the joint distribution, because they are useless for the algorithm, due to its nature of computing contours using points whose Euclidean distances are the highest. From experimental results, we have derived equations (10) and (11) for the calculation of the radius based on the kurtosis and correlation of the mixture signals.

For sub-Gaussian mixtures, the algorithm will try to find the contour of the sensor signal distribution. In this case we determine the exclusion radius as follows:

$$R = \frac{\alpha}{\rho(x)^2 + 0.1} \cdot \overline{x} \quad (10)$$

where α is a constant (experimentally, a value of α=7.5 was applied), $\rho(x)$ is the correlation of the mixtures and

$$\overline{x} = \sqrt{\sum_{j=1}^{N} x(1,j)^2 + x(2,j)^2}\ .$$

For super-Gaussian mixtures (positive kurtosis), the algorithm will search for high density regions of the joint distribution of the mixtures. Thus, the exclusion radius was calculated as:

$$R = 1.5 \cdot \overline{x} \quad (11)$$

In figures 4 and 6, the effect of applying the exclusion radius to a mixture of two voice signals is shown. In this case, 79.2% lay within the exclusion radius and, therefore, they were removed.

3. Simulations and Results.

The new algorithm, firstly named as "LatticeICA", has been tested on various ensembles of artificial sensor signals with an arbitrary number of samples drawn at random from sub- and super-Gaussian distributions like uniform, Gamma, Laplacian and Delta distributions, as well as with real world speech signals. To quantify the performance achieved we calculate both a crosstalking error of the original and recovered source signals as proposed by Amari et al. [2] as well as a component wise crosstalk measure and defined by:

$$E(P) = \sum_{i=1}^{n}\left(\sum_{j=1}^{n}\frac{|p_{ij}|}{\max_k |p_{ik}|} - 1\right) + \sum_{j=1}^{n}\left(\sum_{i=1}^{n}\frac{|p_{ij}|}{\max_k |p_{kj}|} - 1\right) \qquad (12)$$

where P=(p_{ij})= W·A. The parameter MSE (Mean Square Error) measures the similarity of the signals $s_i(t)$ and $y_i(t)$.

3.1 Simulation 1: Uniform noise.

The original source signals are two uniform noises of 110.250 samples. As a representative example we first present a simulation where a lattice of 15 by 15 cells has been calculated within the algorithm to subdivide the space of observations into 225 cells, and threshold (TH) set to 175 samples within each cell. The original and estimated matrices are:

$$A = \begin{pmatrix} 1 & 0.5 \\ -0.7 & 1 \end{pmatrix};\ W^{-1} = \begin{pmatrix} 1 & 0.496 \\ -0.657 & 1 \end{pmatrix} \qquad (13)$$

The Amari performance index for this simulation is $E(W \cdot A) = 0.0703$ .The crosstalk parameters were Crosstalk1(E_{s1})= -53 dB and Crosstalk2(E_{s2})= -29 dB. In Figure 4 it is depicted how the algorithm finds the contour to give the independent components.

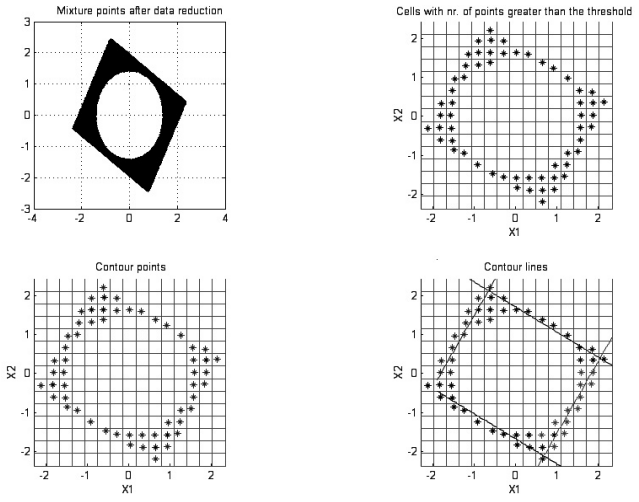

Fig. 4 – Performance of the LatticeICA algorithm for a two uniform noise signals mixture.

3.2 Simulation 2: Speech signals.

Next, the algorithm separate two super-Gaussian signals with a Laplacian distribution and 10000 samples each. The lattice was automatically computed to be 16 rows and 16 columns, using TH= 10. The original and estimated matrices were:

$$A = \begin{pmatrix} 1 & 0.75 \\ 0.6 & 1 \end{pmatrix};\ W^{-1} = \begin{pmatrix} 1 & 0.747 \\ 0.615 & 1 \end{pmatrix} \qquad (14)$$

The joint distribution of the mixtures points out the super-Gaussian nature of the sources (see Figure 5). The matrix performance index for this simulation was $E(W \cdot A) = 1.2931$, with Crosstalk1(E_{s1})=-39 dB and Crosstalk2(E_{s2})=-24 dB. In figure 6 it is ahown how the algorithm searches for the lines of higher density instead of the contour plot).

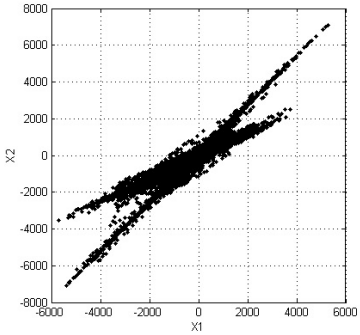

Fig. 5 – Joint distribution of two voice signals.

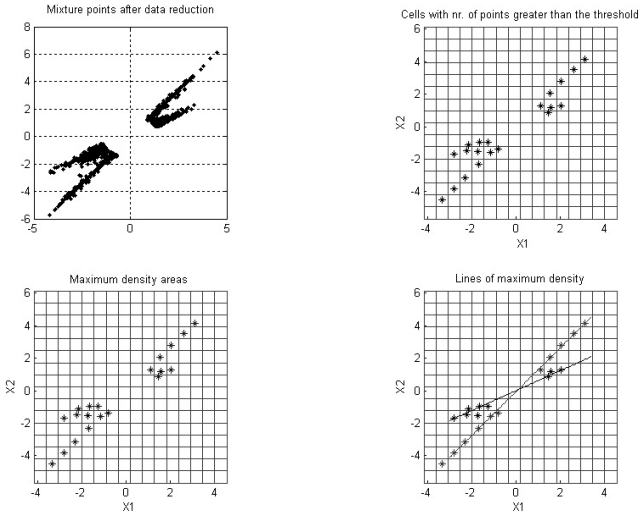

Fig. 6 – Performance of the LatticeICA algorithm for a two real voice signals mixture.

3.3 Simulation 3: Comparison with other algorithms.

In the third simulation we started a more systematic exploration of the algorithm and compared the results to those obtained with two other algorithms, the fastICA Hyvärinen and Oja) and Jade algorithms (Cardoso, 1999). We tried random mixture matrixes over uniform and Laplacian mixtures of 10000 samples, running 100 simulations each time, with automatic parameters. With fastICA the number of bins has been choosen in all cases to be 180. The NRMS (normalized root mean squared error) in each case and the corresponding average

convergence times (Pentium IV 1.5 GHz., 512 MB RAM, Matlab environment) are summarized in Table 1. Although, both FastICA and Jade algorithms globally get better results than LatticeICA in most of the simulations, LatticeICA shows a great performance especially for super-Gaussian mixtures and it outperforms previous geometric algorithms. As a particular advantage of LatticeICA when compared with FastICA and Jade it remains its easy hardware implementation, due to the fact that it only computes simple arithmetic operations. Future enhancements in fine tuning the radius exclusion and adjusting the final separation lines will certainly lead to a better performance.

	NRMS	Speed of convergence (ms.)
Source type: Uniform		
Lattice ICA	0.054	808
FastICA	0.021	501
Jade	0.028	584
Source type: Laplacian		
Lattice ICA	0.034	703
FastICA	0.087	406
Jade	0.009	273

Table 1.Comparison of performance of the new algorithm (shortly named LatticeICA) with FastICA and Jade.

3.4 Simulation 4: Extension to higher dimensionality.

Finally, we show how this algorithm can be extended to higher dimensionality situations by attempting to separate the projections of p mixed signals from $\mathbb{R}^p$ onto $\mathbb{R}^2$. The signals are shown in figures 7 to 9 (with a Laplacian noise, a music source and a speech signal). The original and obtained matrices are:

$$A = \begin{pmatrix} 1 & 0.5 & 0.5 \\ 0.5 & 1 & 0.5 \\ 0.5 & 0.5 & 1 \end{pmatrix} \quad W^{-1} = \begin{pmatrix} 1 & 0.64 & 0.73 \\ 0.45 & 1 & 0.63 \\ 0.46 & 0.47 & 1 \end{pmatrix} \qquad (15)$$

In Figure 8 can be seen the 3-dimensional mixture and the projections in each of the planes which will be the inputs to the algorithm. Figure 9 depicts the separated signals of the proposed LatticeICA algorithm employing radius exclusion and random elimination of points.

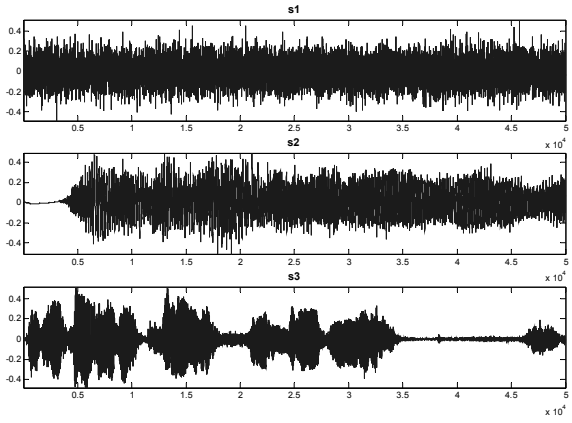

Fig. 7 – Source signals for Simulation 4.

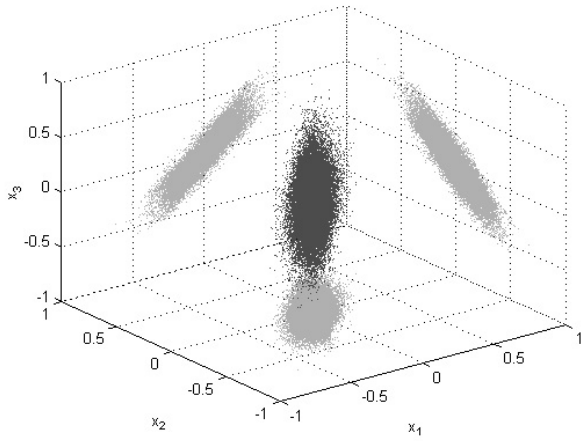

Fig. 8 – Three-dimensional mixture and projections.

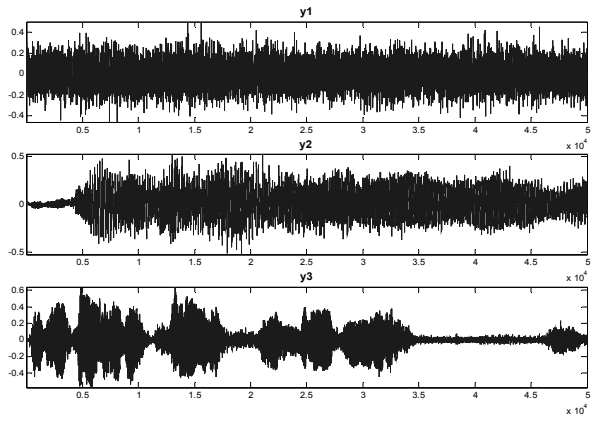

Fig. 9 – Estimated signals for Simulation 4.

Finally, Table 2 shows details about the signals properties, algorithm parameters and simulation results with and without the enhancements proposed. MSE stands for the Mean-Squared of Errors, "Cross" stands for the crosstalk, "Thresh" is the threshold and "Samp" is the number of samples.

FEATURES OF MIXED SIGNALS		
Kurtosis (x_1) = 0.77	Correlation (x_1 , x_2) = 0.85	
Kurtosis (x_2) = 0.17	Correlation (x_1 , x_3) = 0.86	
Kurtosis (x_3) = 1.37	Correlation (x_2 , x_3) = 0.83	
SQUARED ERRORS AND CROSSTALK		
Without data reduction	**With radius exclusion**	**With radius exclusion and random data elimination**
MSE (s_1 , y_1) = 0.147	MSE (s_1 , y_1) = 0.04	MSE (s_1 , y_1) = 0.03
MSE (s_2 , y_2) = 0.095	MSE (s_2 , y_2) = 0.3	MSE (s_2 , y_2) = 0.05
MSE (s_3 , y_3) = 0.055	MSE (s_3 , y_3) = 0.02	MSE (s_3 , y_3) = 0.02
Cros (s_1 , y_1) = - 8dB	Cross (s_1 , y_1) = - 3dB	Cross (s_1 , y_1) = - 4dB
Cros (s_2 , y_2) = 10dB	Cross (s_2 , y_2) = - 5dB	Cross (s_2 , y_2) = - 2dB
Cros (s_3 , y_3) = 12dB	Cross (s_3 , y_3) = -5dB	Cross (s_3 , y_3) = - 6dB
ALGORITHM PARAMETERS		
Grid (x_1, x_2) = 27×27	Grid (x_1, x_2) = 28×28	Grid (x_1, x_2) = 28×28
Thresh (x_1, x_2) = 51	Thresh (x_1, x_2) = 10	Thresh (x_1, x_2) = 5
Time (x_1, x_2)= 39.0 s	Time (x_1, x_2) = 7.2 s	Time (x_1, x_2) = 3.4 s
Samp (x_1,x_2)= 50000	Samp (x_1,x_2) = 10632	Samp (x_1,x_2) = 5440
Grid (x_1, x_3)= 22×22	Grid (x_1, x_3) = 23×23	Grid (x_1, x_3) = 23×23
Thresh (x_1, x_3) = 77	Thresh (x_1, x_3) = 15	Threshold (x_1, x_3) = 8
Time (x_1, x_3)= 26.2 s	Time (x_1, x_3) = 5.1 s	Time (x_1, x_3) = 2.2 s
Samp (x_1,x_3)= 50000	Samp (x_1,x_3) = 10676	Samp (x_1,x_3) = 5338
Grid (x_2, x_3)= 24×24	Grid (x_2, x_3) = 25×25	Grid (x_2, x_3) = 25×25
Thresh (x_2, x_3) = 65	Thresh (x_2, x_3) = 13	Thresh (x_2, x_3) = 6
Time (x_2, x_3)= 31.0 s	Time (x_2, x_3) = 5.75 s	Time (x_2, x_3) = 2.73 s
Samp (x_2,x_3)= 50000	Samp (x_2,x_3) = 10560	Samp (x_2,x_3) = 5333
ORIGINAL AND ESTIMATED MATRICES		
$A = \begin{pmatrix} 1 & 0.5 & 0.5 \\ 0.5 & 1 & 0.5 \\ 0.5 & 0.5 & 1 \end{pmatrix}$		
$W = \begin{pmatrix} 1 & 0.81 & 0.65 \\ 0.29 & 1 & 0.65 \\ 0.45 & 0.48 & 1 \end{pmatrix}$	$W = \begin{pmatrix} 1 & 0.61 & 0.75 \\ 0.77 & 1 & 0.66 \\ 0.48 & 0.55 & 1 \end{pmatrix}$	$W = \begin{pmatrix} 1 & 0.64 & 0.73 \\ 0.45 & 1 & 0.63 \\ 0.46 & 0.47 & 1 \end{pmatrix}$
Amari Index = 2.65	Amari Index = 2.52	Amari Index = 1.73

Table 2. Parameters, Errors and Matrices for Simulation 4

4. CONCLUSIONS.

We have developed a new geometry-based method for blind separation of sources which greatly reduces the complexity and computational load inherent in the standard geometric ICA algorithms. This new algorithm is based on a tessellation of the input space where in each cell a code book vector is determined to represent the center of gravity of the local distribution of sample vectors. Depending on the type of distribution, either sub- or super-Gaussian, the slopes of the border lines or the diagonals are determined to obtain the coefficients of the estimated mixing matrix *W*. The method lends itself for an easy hardware implementation and is also very intuitive in terms of computer applications. Furthermore, this method could be used to detect the perimeter or outlines in simple two-dimensional figures. In the future we will intend to implement this method for more than two signals without using projections but working in the p-dimensional space.

ACKNOWLEDGEMENT

This work has been supported by the Spanish CICYT Projects TIC2001-2845 "PROBIOCOM" (Procedures for Biomedical and Communications Source Separation) and TIC2000-1348 (Hybrid Procedures for Cluster Parallel Optimization).

REFERENCES

[1] M. R. Álvarez, C. G. Puntonet, I. Rojas, Separation of Sources based on the Partitioning of the Space of Observations, *LNCS* 2085, 2001, 762 – 769.

[2] S.Amari, A.Cichocki, H.H.Yang, A new learning algorithm for blind signal separation*, Proc. NIPS'96*, 8, 1996, 757-763

[3] A.J. Bell and T. J. Sejnowski, An information-maximisation approach to blind separation and blind deconvolution, *Neural Computation*, 7, 1995, 1129-1159.

[4] J-F. Cardoso. High-order contrasts for independent component analysis. *Neural Computation*, 11(1), 157-192, 1999.

[5] A. Hyvärinen, J. Karhunen, E. Oja, Independent Component Analysis, *Wiley & Sons*, New York, 2001

[6] A.Jung, F.J.Theis, C.G.Puntonet, E.W.Lang, FASTGEO: A histogram based approach to linear geometric ICA, *Proc. ICA'01*, 2001, 349-354

[7] C. Jutten, J. Hérault, P. Comon, E. Sorouchiary, Blind separation of sources, Parts I, II, III. *Signal Processing*, 24 (1), 1991, 1-29.

[8] C.G.Puntonet, A.Prieto, Neural net approach for blind separation of sources based on geometric properties, *Neurocomputing*, 18, 1998, 141-164

[9] F.J.Theis, A.Jung, E.W.Lang, C.G.Puntonet, Linear Geometric ICA: Fundamentals and Algorithms, *Neural Computation,* 2002, in print.